Presented at the Association of Computing Machinery (ACM) Symposium on Applied Computing SAC 2002, March 10 - 13, 2002 Madrid, Spain

Cathedrals, Libraries and Bazaars

by Ken Krechmer, Fellow

International Center for Standards Research University of Colorado at Boulder

Communications Standards Review
757 Greer Road
Palo Alto, CA USA 94303-3024
+1 650 856-8836
http://www.csrstds.com
krechmer@csrstds.com

Abstract

The open source and open standards movements are sometimes confused and sometimes clash. This seems unfortunate as they are both vital to the growing acceptance of technology and have much to offer each other. Expanding on the metaphors offered by Eric Raymond in his paper, The Cathedral and the Bazaar, this essay describes the similarities and differences between the concepts of open source and open standards. Openness has a very different meaning as applied to software programs or technical standards. As this essay attempts to explain, the open source and open standards movements are most closely related by their desire to allow competition to thrive based on merit, not market dominance.

In an excellent paper, The Cathedral and the Bazaar [1], Eric Raymond proposes that open source code makes possible an open market (a bazaar) which then enriches open source programs. Programmers implement new capabilities in open source programs driven by their desire to improve the software and showcase their skills. He suggests that proprietary or closed source software programs are like a cathedral, useful and even inspiring when created, but too often left behind without the continuing enrichment of new ideas and capabilities.

This essay explores the similarities and the differences between open source programs and open technical standards by expanding on Raymond's architectural metaphor of cathedral and bazaars. A public library offers another architectural model which is not as monolithic as a cathedral nor as flexible as a bazaar. Items in a library are available for public use, but are not available to be readily modified. The addition and removal of items in a public library is not usually a public process.

Cathedrals, libraries and bazaars as they are used here describe powerful paradigms. Cathedrals represent the stability and strength society finds in centralized structures and rules. Libraries represent the value to society of vetted and maintained knowledge, publicly available. Bazaars represent a marketplace full of new ideas, the freedom to change and evolve. While all processes must evolve or cease, the most desirable rate of evolution is a subject of considerable debate. The library metaphor suggests a moderate pace of change, which may be most broadly acceptable. As this paper will explain, the pace and process of change in open software programs and open technical standards is currently quite different, for practical reasons.

Understanding the paradigms of our technical age is not easy. Cathedrals and libraries are both the repositories of rules. The rules from the cathedral are only available to the initiated while the rules from the library are available to all. Since both libraries and cathedrals store rules, some consider technical standards, a type of rules, as more like a cathedral than a library. This is understandable considering that governments may enforce technical standards via a regulation or legal decision, adding the force of law to the use of what was a voluntary technical standard. However, regulations are created independently of a voluntary, consensus technical standard. A governmental or legal decision does not impugn the voluntary and consensus process used to create the standard. This essay considers voluntary (created without coercion, even though their use may be enforced), consensus technical standards as independent of any structure that enforces them.

Once, IBM software programs were considered a significant repository of stability, like a cathedral. Even today, many user software programs are very difficult to change or adapt to new requirements. This is not generally desirable. A new compromise is needed between the stability of the cathedral and adaptability of the bazaar. Software programs, operating in a computer with changeable memory may be modified at will, if the necessary technical skill is available. The open source software movement strives to allow the most desired modifications of a software program to be created rapidly (based on the availability of skilled and interested programmers). Raymond proposes that open source programs are like the bazaar (open and available). In comparison to the cathedral in his metaphor, he is certainly correct.

However, Raymond's metaphor is expanded in this paper to include the library. Raymond proposes that open source programs are like a bazaar and to a programmer this appears to be true. But to a user, not trained as a programmer, open source programs seem to be more like a public library - available for any user, but changeable only by programmers who understand how to do so. The library metaphor also applies to technical standards. The library thus becomes the metaphor for both the technical standards and computer programs used to support an increasingly technical world. Now that we see both open source and open standards in the same light, it is appropriate to examine the differences between them to determine how the values of each can improve the other.

The differences between standards and software programs

The term "voluntary" referring to standards is easily taken as similar to the term "open" applied to source programming. This is too simple a view. A voluntary, consensus technical standard describes an implementation or use of the information it contains. When a specific software program is used as a test bed for testing, then the specific program in the test bed takes on the characteristics of a technical standard. It is also true that an ASN.1 [2] representation of a technical standard (usually defining a communications protocol) may be compiled into an operational software program. While both open source programs and open standards may be seen as knowledge available from a library, these conversions only illuminate the fact that standards and programs are different.

Raymond, as he explains in The Cathedral and the Bazaar, is (among other roles) the development coordinator of an open source program called "fetchmail." This is a widely used program in e-mail systems. As such, it is considered by many other programmers to be a "standard." The programmers' use of the term standard here is an acknowledgement of the widespread use of fetchmail, and a compliment. But it is also the beginning of some confusion.

A technical standard may be defined as a codification, describing an implementation or procedure, which is used for the purpose of comparison [3]. In terms of formal technical standards, the use of fetchmail represents a convention, as there is no formal written document (codification) that specifies the use of fetchmail. Conventions may be ignored or changed when a better solution is identified. This continuing evolution to a better solution, C. Shirky [4] explains, is an important feature of open source programs. In a

later publication, Raymond condenses the definition of open source to: Open source is software that is freely redistributable and can readily be evolved and modified to fit changing needs [5] - quite a different definition than a technical standard.

Further confusion is added because the term "de facto standard" (a "standard" by virtue of its use) comes very close to the concept of a convention. A formal technical standard is not a convention. It is maintained until the society that created it formally agrees it is no longer of value [6]. The specific copy of fetchmail that Raymond maintains might be called a "de facto standard," as it is the measure of all other implementations of fetchmail, but it is only a desirable convention. This examination of definitions and words exposes some of the differences between open source and open standards. Over time it is possible that open source programming may become more formalized and open standards more adaptable, but there will still be substantial differences between the two.

What does "open" mean?

The bazaar metaphor offers one image of openness: that of a vibrant marketplace. The adjective "open" means different things when applied to the concepts of technical standards and software development. In the open source movement, openness implies an ability to access and change the source code, at any time, to support a desired capability. In terms of open standards, the most widely agreed use of openness implies a willingness to accept external input during the standards development process. There is some similarity between these concepts, but they are not the same. The companies that have the most to gain from closed source programs and closed specifications are sometimes the quickest to misuse the terms open source and open standards [7]. Definitions are important here. There is, apparently, a very profitable business in calling closed specifications and closed programs "open."

The most obvious difference between an open source program and an open standard is temporal. A standard is a codification that a society (collection of users and implementers) wishes to maintain unchanged for a span of time. Openness is supported during the standards development process, upon completion of which, the standard is considered stable and thus no longer open to rapid change. Raymond suggests in open source lesson 7, "Release early. Release often." In standards development, the goal is to create stability. In open source program development, the goal is to support change, as the development process is considered on-going. These goals are different and the way to achieve them is different.

Even the word "open" means something very different in open source than in open standards. In open source it refers to the ability of programmers to access source code to make changes in the program. Such programmers are a small and select group of people. "Open" in relation to standards has many meanings, not all agreed, but today predominately refers to the ability of anyone to propose a function in the standard-to-be, and receive due consideration.

The many concepts of "open source" and "open standards"

Programs consist of instructions to computers; such instructions must change and adapt, otherwise fixed logic would suffice. In many systems, the ability of the software programs to change and adapt is only available to the system developer and not to the system implementers, knowledgeable user or system owner, even when they are supported by skilled programmers. This is one of the issues that the open source movement wishes to change.

Standards are developed because some forms of knowledge are too important to a society to change quickly. Measuring systems, currency, building codes, safety standards, telephone systems, protocols (e.g.,TCP/IP), etc. add value or reduce risk to society when they are maintained for longer spans of time.

The lengthy transition in the Internet Engineering Task Force (IETF) from Internet Protocol version4 to Internet Protocol version6 is an indication, among other things, of the importance to society of a stable version of the Internet Protocol as a standard.

Technical standards are created by many different standards development organizations (SDOs), in every field. In the communications field, SDOs include the IETF, International Telecommunications Union, ATIS Committee T1, DSL Forum, ATM Forum, Frame Relay Forum, WAP Forum, and others. With the growth in technical standards and the decline in government regulation of standards, the number of such standards development organizations has become very large and it is confusing to identify which SDO, if any, offers open standards. It is even confusing to identify what an open standard is.

Open source

Both open standards and open source encompass a broad range of concepts. "The Cathedral and the Bazaar" describes open source programming in 19 lessons. Below are extracted ten lessons that may apply to standards development as well as software program development. In the following lessons just replace the word "software" with "standard," and "programmer" with "standards developer" to see how they may apply to standards development.

1. Every good work of software starts by scratching a programmer's personal itch.
2. Good programmers know what to write. Great ones know what to rewrite (and reuse).
3. Plan to throw one (software version) away; you will, anyhow.
6. Treating your users as co-developers is your least-hassle route to rapid software improvement and effective debugging
8. Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone.
10. If you treat your beta-testers as if they're your most valuable resource, they will respond by becoming your most valuable resource.
12. Often, the most striking and innovative solutions come from realizing that your concept of the problem was wrong.
13. Perfection (in software design) is achieved not when there is nothing more to add, but rather when there is nothing more to take away.
14. Any program should be useful in the expected way, but a truly great program lends itself to uses you never expected.
19. Provided the development coordinator (programmer) has a medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one.

Lessons 1, 2, 8, 10, and 19 could almost be a description of how the Internet Engineering Task Force (IETF) operates their standards development process. The IETF standards and standardization process have been very successful. IETF standards are in wide use with broad attendance and activity in the IETF standardization meetings. So Raymond's lessons may offer helpful advice for other standardization organizations.

It is not possible to apply lessons 6 and 8 to all standards development because there is a temporal problem comparing software programs (implementations) to standards (codifications). Standards are codified in three time periods relative to implementations of the standard. Anticipatory standards, such as V.90 modems or ISO 9000 (quality procedures), lead implementations. The IETF standards may be seen as participatory standards, where two implementations are required for IETF standardization to occur. Responsive standards occur when a proprietary specification has become widely implemented and then is formally standardized. Lessons 6 and 8 seem to apply to participatory standards and responsive standards but less so to anticipatory

standards.

Open standards

Similar to open source programs, open standards is a changing concept, molding itself to the evolving needs of an open, consensus-based society. Currently ten principles are considered, at least by some, to constitute the principles of open standards [8].

1. Openness - all stakeholders may participate in the standards development process.
2. Consensus - all interests are discussed and agreement found, no domination.
3. Due Process - balloting and an appeals process may be used to find resolution.
4. Open IPR - holders of Intellectual Property Rights (IPR) must identify themselves during the standards development process.
5. Open World - same standard for the same function, world-wide.
6. Open Access - all may access committee documents, drafts and completed standards.
7. Open Meeting - all may participate in standards development meetings
8. On-going Support - standards are supported until user interest ceases rather than when provider interest declines.
9. Open Interfaces - interfaces allow additional functions, public or proprietary.
10. Open Use - low or no charge for IPR necessary to implement an accredited standard.

The first four principles are widely acknowledged, the fifth principle is agreed but not required by the American National Standards Institute (and many other standards development organizations), while the last five are just beginning to be considered by formal standards organizations. As example, Open Use (10) supports the free use of an open standard. This is part of the definition of open source, but a far-off ideal for open standards. Open standards is an evolving concept, reaching towards a distant perfection.

As these two lists point out, the goals and functions associated with open source programming and open technical standards are quite different. But understanding the goals of one may be quite useful to enhancing openness in the other.

Open source programming supports continuing change and enhancements based on the market's needs but is still learning the value of standard interfaces. And the concepts of consensus, due process and open IPR are still emerging in relation to open source programming. The standards world has been taught these concepts well, by expensive lawyers, over the past 100 years. Conversely, open standards now only support open requirements followed by a period of stability. In open standards the concept of free IPR is very remote, and easy adaptability appears to be in conflict with the needs for stable standards. However, new approaches to offering adaptability in technical standards are beginning to emerge [9].

The open source movement, in developing and describing its open programming process, illuminates the value of continuous adaptability and its path to achieve such adaptability. Conversely, the lengthy process towards openness from a standards perspective offers a more complete view of openness than has been developed by the open source movement. The concepts of open source and open standards have definite value to offer each other. In the future it appears possible to combine the adaptability of open source programs with the stability of open technical standards. Such a combination may support significant new technical advancements.

Common adversaries

The concepts of open source and open standards currently have a common adversary in the marketplace -

Microsoft. Since Microsoft attempts to neutralize open source and open standards independently, understanding each concept as related, but independent, may allow the efforts to balance Microsoft's market dominance to develop on two fronts. Microsoft is certainly not the only company that desires to control its markets beyond the value offered to their customers. The desire for such control is quite understandable in a capitalist system. It is the unrestrained use of such control that must be tempered by open standards, competition and government action (if necessary). Open standards allow other companies to compete with Microsoft based on the merits of their respective products [10]. Open source programs (e.g., Linux) provide direct competition to Microsoft products.

The expanding competition from open source programs, if they also support open standards, offers one way to pressure Microsoft and others to support open standards as well. It is possible to provide open source and yet support standard interfaces if the open source movement can appreciate the value of standard interfaces. Then open standards level the playing field for all competitors.

Common goals

The cathedral, the library and the bazaar are as vital to society as they ever were, but the emphasis society places on each paradigm has shifted. "First world" society has given up much of the trust in the centralized strength and stability of governments and large companies (in this paper's metaphor, cathedrals) that it maintained up to the middle of the 20th century. Now libraries (e.g., the Internet), as repositories of software programs and technical standards, provide the basic knowledge necessary to trade, operate and communicate. With this new focus, society is learning to identify which programs and technical standards are open and public and what advantages this confers.

Libraries are where both programs and standards reside, but finding open source programs and open technical standards is becoming more complex. Manufacturers, developers and service providers will, for commercial reasons, attempt to control their products and services and avoid using the libraries that provide open standards and open source.

This paper has explained the similarities and the differences between open source and open standards in an attempt to show how these two related but distinct movements can gain by better understanding each other. The open source and open standards movements are most closely related by their desire to allow competition to thrive based on merit, not market dominance. The proponents of both open standards and open source have the Sisiphyan task of explaining what is open and what is not, what that means and where to find it. This ceaseless task is vital if the bazaar, and the tantalizing freedom it offers to rapidly change and evolve, is to be more than a dream.

Ken Krechmer
Communications Standards Review
757 Greer Road
Palo Alto, California 94303-3024 USA

VOICE: +1 650 856-8836
e-mail: krechmer at csrstds.com

Return to the directory of Ken's lectures, published articles and book reviews

---

Footnotes

[1] Eric S. Raymond, The Cathedral and the Bazaar, August 24, 2000, http://www.redhat.com/support/wpapers/community/cathedral/whitepaper_cathedral-2.html
Return to text

[2] Abstract Symbol Notation, a formal logical language.
Return to text

[3] Ken Krechmer and Elaine Baskin, Standards, Information, Communications; Proceedings of SIIT2001, October, 2001. http://www.csrstds.com/siit2001.html
Return to text

[4] "Only solutions that produce partial results when partially implemented can succeed." Clay Shirky, In Praise of Evolvable Systems, http://www.shirky.com/OpenSource/evolve.html.
Return to text

[5] Eric Steven Raymond, Homesteading the Noosphere, section 2, August 25, 2000. http://www.tuxedo.org/~esr/writings/cathedral-bazaar/homesteading/
Return to text

[6] American National Standards Institute, Due process and criteria for approval and withdrawal of American National Standards, http://www.ansi.org/public/library/revise/procedure_updates/due_proc1a.html#criteria
Return to text

[7] Compete, Don't Delete, *The Economist*, June 11, 1998. Defending Microsoft, Bill Gates used the term "open standards" five times referring to Microsoft proprietary specifications.
Return to text

[8] Ken Krechmer, The Principles of Open Standards, *Standards Engineering*, Vol. 50, No. 6, November/December, 1998, http://www.csrstds.com/openstds-old.html
Return to text

[9] Ken Krechmer & Elaine Baskin, The Fundamental Nature of Standards: Technical Perspective, *IEEE Communications Magazine*, June, 2000, Vol 38 #6, http://www.csrstds.com/fundtec.html
Return to text

[10] Ken Krechmer & Elaine Baskin, The Microsoft Anti-Trust Litigation: the Case for Standards, first place winner World Standards Day 2000, Society for Engineering Standards, http://www.ses-standards.org/library/krechmerbaskin.pdf.
Return to text

This page was last updated October 12, 2007.

Return to CSR Home Page